Burak Tekdamar
161044115
CSE 344 HW5 REPORT

1 Overview

First of all, to solve the problem, I read the parameters given by the user one by one with the getopt() function and assigned them to the variables. I checked whether the parameters given by the user are valid. I returned an error message if there is an invalid parameter or a missing parameter. Then I created a struct called arrayBlock for all the arrays I will use in the program. I set NULL as initial value to all of these arrays. Then I opened the input and output files. After reading the input files and filling the arrays, I created the threads and joined them. After the threads finished the processes, I printed the results I obtained to the output file and freed all the open resources. I wrote a handler to caught the SIGINT signal, thus freeing up all the resources that were open until the signal was caught.

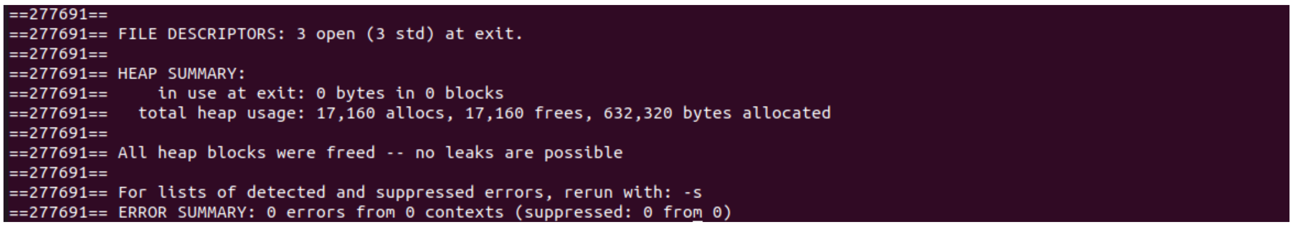
2 How did I solve this

First of all, I kept a pointer named threads for the threads I will use. I converted this pointer into a dynamic array with the size of the m value entered by the user. then I created 2D arrays for A and C matrices and 3D arrays for B matrix. I filled the matrix A with the values I read from the first input file and the matrix B with the values I read from the second input file. The first dimension of matrix B represents which thread the values in that array belong to. The second dimension of the B matrix is 2^n and the third dimension is 2^n/m.

I kept a counter value while reading the files. I increased the counter value by one for each character I read from the files. If there are not 2^n characters in a file, the program prints an error message and terminates. I create threads after the files are read and the arrays are filled. I used the "synchronization barrier with N threads" algorithm found in the week10.pdf file to wait for all threads to occur. After all threads are created, the condition variable is broadcast and all threads continue to work to calculate their own C values. After all threads calculate their C matrices, they write their matrices into the C matrix, which I keep in the global. After all the threads have finished the C calculation, they go to the 2nd part for the dft calculation. I also created a barrier structure for the 2nd part. Since all threads reach part2, the condition variable is broadcast again and they start dft calculations. Each thread calculates the dft matrix of its own section, just like the C calculation. After all threads have completed the dft calculation, the threads are terminated. After the threads are terminated, I print all the data I have obtained to the output file given by the user. After the printing process is finished, I free all open resources.

As can be seen from the terminal outputs on the next page, as the number of threads increases, the load per thread and the C matrix and dft calculation times of the threads decrease. When the same files are run with 8 threads, they can be finished faster than if they are run with 4 threads.

Valgrind Result:

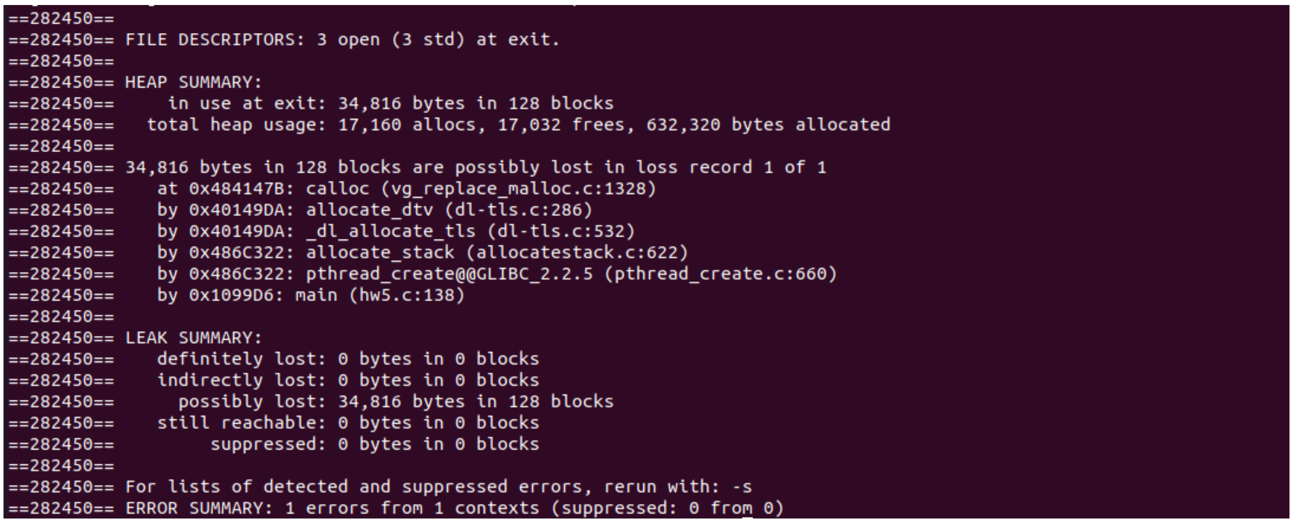


```
==277691==
==277691== FILE DESCRIPTORS: 3 open (3 std) at exit.
==277691==
==277691== HEAP SUMMARY:
==277691==     in use at exit: 0 bytes in 0 blocks
==277691==   total heap usage: 17,160 allocs, 17,160 frees, 632,320 bytes allocated
==277691==
==277691== All heap blocks were freed -- no leaks are possible
==277691==
==277691== For lists of detected and suppressed errors, rerun with: -s
==277691== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Running hw5 program:

./hw5 -i filePath1 -j filePath2 -o output -n charecterNum -m threadNum

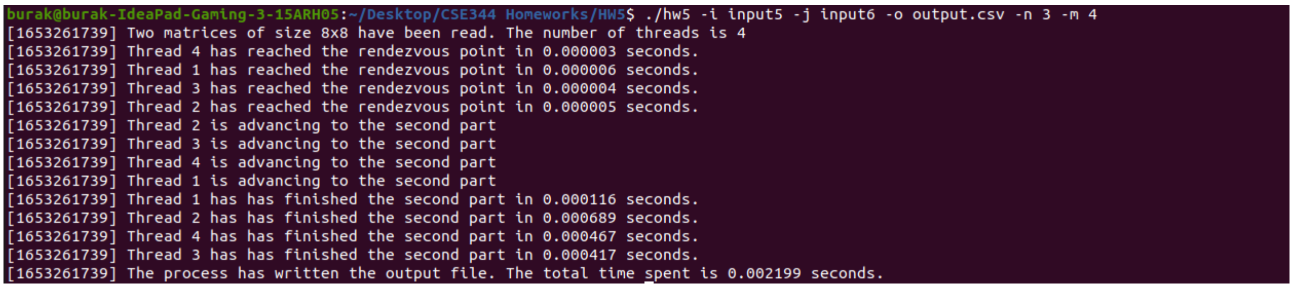
I wrote a handler to catch the SIGINT signal. When the user presses the CTRL+C key, all open resources created so far are freed. When CTRL C is pressed, a possible leak appears in valgrind as much as 272 byte * threadNumber due to the pthread_create function.



```
==282450==
==282450== FILE DESCRIPTORS: 3 open (3 std) at exit.
==282450==
==282450== HEAP SUMMARY:
==282450==     in use at exit: 34,816 bytes in 128 blocks
==282450==   total heap usage: 17,160 allocs, 17,032 frees, 632,320 bytes allocated
==282450==
==282450== 34,816 bytes in 128 blocks are possibly lost in loss record 1 of 1
==282450==    at 0x484147B: calloc (vg_replace_malloc.c:1328)
==282450==    by 0x40149DA: allocate_dtv (dl-tls.c:286)
==282450==    by 0x40149DA: _dl_allocate_tls (dl-tls.c:532)
==282450==    by 0x486C322: allocate_stack (allocatestack.c:622)
==282450==    by 0x486C322: pthread_create@@GLIBC_2.2.5 (pthread_create.c:660)
==282450==    by 0x1099D6: main (hw5.c:138)
==282450==
==282450== LEAK SUMMARY:
==282450==    definitely lost: 0 bytes in 0 blocks
==282450==    indirectly lost: 0 bytes in 0 blocks
==282450==      possibly lost: 34,816 bytes in 128 blocks
==282450==    still reachable: 0 bytes in 0 blocks
==282450==         suppressed: 0 bytes in 0 blocks
==282450==
==282450== For lists of detected and suppressed errors, rerun with: -s
==282450== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
```

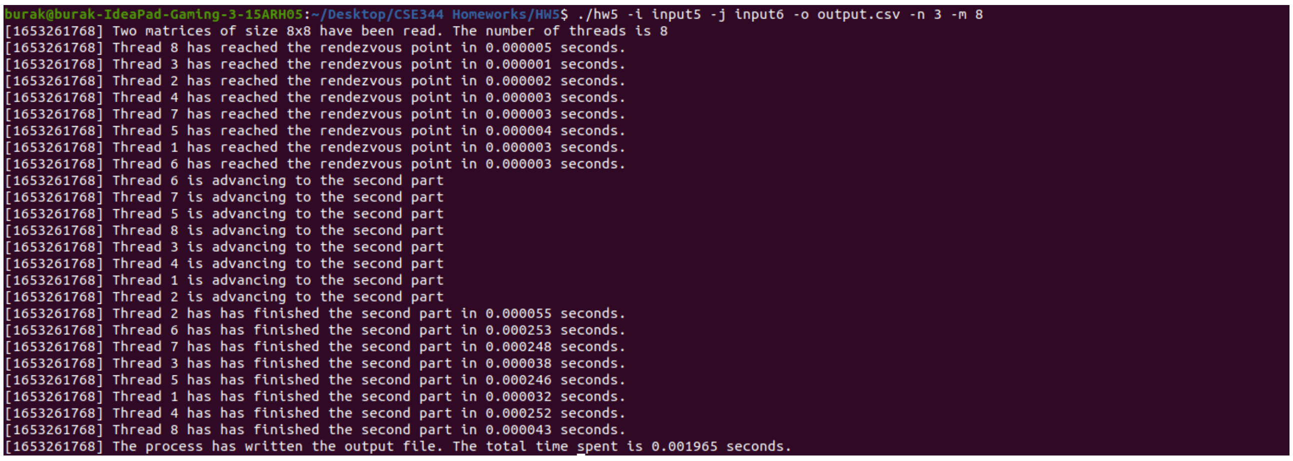
OUTPUTS

Terminal outputs



```
burak@burak-IdeaPad-Gaming-3-15ARH05:~/Desktop/CSE344 Homeworks/HW5$ ./hw5 -i input5 -j input6 -o output.csv -n 3 -m 4
[1653261739] Two matrices of size 8x8 have been read. The number of threads is 4
[1653261739] Thread 4 has reached the rendezvous point in 0.000003 seconds.
[1653261739] Thread 1 has reached the rendezvous point in 0.000006 seconds.
[1653261739] Thread 3 has reached the rendezvous point in 0.000004 seconds.
[1653261739] Thread 2 has reached the rendezvous point in 0.000005 seconds.
[1653261739] Thread 2 is advancing to the second part
[1653261739] Thread 3 is advancing to the second part
[1653261739] Thread 4 is advancing to the second part
[1653261739] Thread 1 is advancing to the second part
[1653261739] Thread 1 has has finished the second part in 0.000116 seconds.
[1653261739] Thread 2 has has finished the second part in 0.000689 seconds.
[1653261739] Thread 4 has has finished the second part in 0.000467 seconds.
[1653261739] Thread 3 has has finished the second part in 0.000417 seconds.
[1653261739] The process has written the output file. The total time spent is 0.002199 seconds.
```

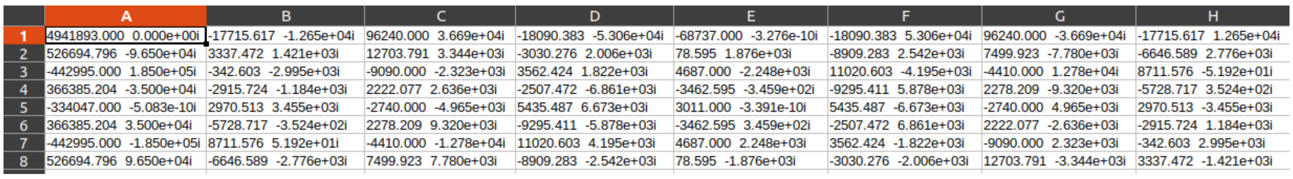
m = 4



```
burak@burak-IdeaPad-Gaming-3-15ARH05:~/Desktop/CSE344 Homeworks/HW5$ ./hw5 -i input5 -j input6 -o output.csv -n 3 -m 8
[1653261768] Two matrices of size 8x8 have been read. The number of threads is 8
[1653261768] Thread 8 has reached the rendezvous point in 0.000005 seconds.
[1653261768] Thread 3 has reached the rendezvous point in 0.000001 seconds.
[1653261768] Thread 2 has reached the rendezvous point in 0.000002 seconds.
[1653261768] Thread 4 has reached the rendezvous point in 0.000003 seconds.
[1653261768] Thread 7 has reached the rendezvous point in 0.000003 seconds.
[1653261768] Thread 5 has reached the rendezvous point in 0.000004 seconds.
[1653261768] Thread 1 has reached the rendezvous point in 0.000003 seconds.
[1653261768] Thread 6 has reached the rendezvous point in 0.000003 seconds.
[1653261768] Thread 6 is advancing to the second part
[1653261768] Thread 7 is advancing to the second part
[1653261768] Thread 5 is advancing to the second part
[1653261768] Thread 8 is advancing to the second part
[1653261768] Thread 3 is advancing to the second part
[1653261768] Thread 4 is advancing to the second part
[1653261768] Thread 1 is advancing to the second part
[1653261768] Thread 2 is advancing to the second part
[1653261768] Thread 2 has has finished the second part in 0.000055 seconds.
[1653261768] Thread 6 has has finished the second part in 0.000253 seconds.
[1653261768] Thread 7 has has finished the second part in 0.000248 seconds.
[1653261768] Thread 3 has has finished the second part in 0.000038 seconds.
[1653261768] Thread 5 has has finished the second part in 0.000246 seconds.
[1653261768] Thread 1 has has finished the second part in 0.000032 seconds.
[1653261768] Thread 4 has has finished the second part in 0.000252 seconds.
[1653261768] Thread 8 has has finished the second part in 0.000043 seconds.
[1653261768] The process has written the output file. The total time spent is 0.001965 seconds.
```

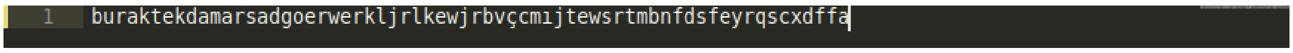
m = 8

Output file



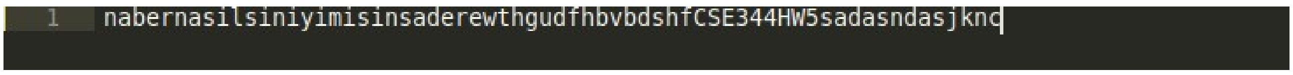
	A	B	C	D	E	F	G	H
1	4941893.000 0.000e+00i	-17715.617 -1.265e+04i	96240.000 3.669e+04i	-18090.383 -5.306e+04i	-68737.000 -3.276e-10i	-18090.383 5.306e+04i	96240.000 -3.669e+04i	-17715.617 1.265e+04i
2	526694.796 -9.650e+04i	3337.472 1.421e+03i	12703.791 3.344e+03i	-3030.276 2.006e+03i	78.595 1.876e+03i	-8909.283 2.542e+03i	7499.923 -7.780e+03i	-6646.589 2.776e+03i
3	-442995.000 1.850e+05i	-342.603 -2.995e+03i	-9090.000 -2.323e+03i	3562.424 1.822e+03i	4687.000 -2.248e+03i	11020.603 -4.195e+03i	-4410.000 1.278e+04i	8711.576 -5.192e+01i
4	366385.204 -3.500e+04i	-2915.724 -1.184e+03i	2222.077 2.636e+03i	-2507.472 -6.861e+03i	-3462.595 -3.459e+02i	-9295.411 5.878e+03i	2278.209 -9.320e+03i	-5728.717 3.524e+02i
5	-334047.000 -5.083e-10i	2970.513 3.455e+03i	-2740.000 -4.965e+03i	5435.487 6.673e+03i	3011.000 -3.391e-10i	5435.487 -6.673e+03i	-2740.000 4.965e+03i	2970.513 -3.455e+03i
6	366385.204 3.500e+04i	-5728.717 -3.524e+02i	2278.209 9.320e+03i	-9295.411 -5.878e+03i	-3462.595 3.459e+02i	-2507.472 6.861e+03i	2222.077 -2.636e+03i	-2915.724 1.184e+03i
7	-442995.000 -1.850e+05i	8711.576 5.192e+01i	-4410.000 -1.278e+04i	11020.603 4.195e+03i	4687.000 2.248e+03i	3562.424 -1.822e+03i	-9090.000 2.323e+03i	-342.603 2.995e+03i
8	526694.796 9.650e+04i	-6646.589 -2.776e+03i	7499.923 7.780e+03i	-8909.283 -2.542e+03i	78.595 -1.876e+03i	-3030.276 -2.006e+03i	12703.791 -3.344e+03i	3337.472 -1.421e+03i

Input files



```
buraktekdamarsadgoerwerkljrlkewjrbvçcmıjtewsrtmbnfdsfeyrqscxdffa
```

input1



```
nabernasilsiniyimisinsaderewthgudfhbvbdshfCSE344HW5sadasndasjknc
```

input2